Enhancing Visual Representation with Textual Semantics: Textual Semantics-Powered Prototypes for Heterogeneous Federated Learning (Supplementary Material)

Xinghao Wu[1] Jianwei Niu[1,2] Xuefeng Liu[1,2*] Guogang Zhu[1]
Jiayuan Zhang[1] Shaojie Tang[3] Wei Chen[1]
[1]State Key Laboratory of Virtual Reality Technology and Systems,
School of Computer Science and Engineering, Beihang University, Beijing, China
[2]Zhongguancun Laboratory, Beijing, China
[3]Center for AI Business Innovation, Department of Management Science and Systems,
School of Management, University at Buffalo, USA
{wuxinghao, niujianwei, liu_xuefeng, buaa_zgg, zhangjiayuan, chenweibuaa}@buaa.edu.cn
shaojiet@buffalo.edu

A. Pseudo-code of FedTSP

The pseudo-code of FedTSP is summarized in Algorithm 1.

Algorithm 1 FedTSP

Input: Each client's initial personalized models $w_i^{(0)}$; The initial trainable prompts $\{v_c^{(0)}\}_{c=1}^{C}$; Server-side pretrained language model (PLM); Server-side large language model (LLM) or LLM API; Client Number N; Total round T; Epochs of local training round E_c; Epochs of server training round E_s.
Output: Personalized model $w_i^{(T)}$ for each client.
Server-side:
Generating fine-grained descriptions for each class and construct prompts $\{p_c\}_{c=1}^{C}$ using Eq. (3).
Mapping $\{p_c\}_{c=1}^{C}$ to $\{\tilde{p}_c\}_{c=1}^{C}$ using the PLM embedding layer.
Client-side:
for $i = 1$ to N **in parallel do**
  Calculating local prototypes by Eq. (5) and send them to the server.
end for
for $t = 0$ to $T - 1$ **do**
  Server-side:
  Aggregating a set of local prototypes to obtain global image prototypes $\{\overline{P}_c^{\mathbf{I}}\}_{c=1}^{C}$.
  Inserting trainable prompts $\{v_c^{(t)}\}_{c=1}^{C}$ into $\{\tilde{p}_c\}_{c=1}^{C}$ to obtain $\{\hat{p}_c^{(t)}\}_{c=1}^{C}$ by Eq. (4).
  Updating $\{v_c^{(t)}\}_{c=1}^{C}$ by Eq. (7) for E_s epochs to obtain $\{v_c^{(t+1)}\}_{c=1}^{C}$.
  Calculating text prototypes $\{P_c^{\mathbf{T}}\}_{c=1}^{C}$ and send them to each client.
  Client-side:
  for $i = 1$ to N **in parallel do**
    Receiving text prototypes and updating local model w_i^t for E_c epochs by Eq. (2) to obtain w_i^{t+1}.
    Calculating local prototypes by Eq. (5) and send them to the server.
  end for
end for

*Corresponding author

B. Training Cost Analysis

In FL, particularly in cross-device FL scenarios, clients typically possess limited computational, communication, and storage resources, posing challenges for deploying resource-intensive algorithms. Here, we analyze the training cost associated with our proposed FedTSP.

Communication Cost. During each training round, clients upload only their image feature centers $P_i^{\mathbf{I}}$, while the server dispatches text prototypes $\overline{P}^{\mathbf{T}}$ to the clients. This results in both uplink and downlink communication costs of $C \times d$, consistent with existing prototype-based methods. Compared to approaches that transmit model parameters, the reduced number of prototype parameters enhances communication efficiency.

Computation Cost. On the client side, FedTSP introduces no additional computation beyond existing prototype-based methods. Clients update their models using Eq. (2) and compute local feature centers via Eq. (5).

On the server side, FedTSP employs a PLM and accesses an LLM through an external API. The LLM API is called only once before training to generate class descriptions, introducing negligible computational and communication overhead. The PLM remains frozen during training, and only a small set of prompts ($C \times m \times d'$) are updated. This overhead does not scale with the number of clients and is easily handled by typical server resources. If API access is unavailable, a lightweight locally deployed LLM can also be used to generate descriptions, which slightly increases memory and computational requirements but does not affect the overall scalability of the framework.

Memory Cost. On the client side, FedTSP maintains the same memory footprint as existing prototype-based methods, which store the local model w_i, local feature centers $P_i^{\mathbf{I}}$, and received text prototypes $\overline{P}^{\mathbf{T}}$.

On the server side, the PLM contributes minimal additional memory usage (*e.g.*, about 0.44 GB for BERT [1]). When using an LLM API, no local model is stored, thus avoiding the memory cost of large-scale language models. If a local LLM is deployed, the memory demand depends on its scale (*e.g.*, approximately 24 GB for LLaMA-7B [17]), which remains practical for modern server infrastructures.

Overall, FedTSP achieves efficient communication and maintains manageable computation and memory requirements, making it suitable for deployment in resource-constrained FL environments.

C. Experimental Setup

C.1. Datasets

We evaluate the effectiveness of FedTSP on three benchmark image classification datasets: CIFAR-10 [5], CIFAR-100 [4], and Tiny ImageNet [6]. The datasets are evenly distributed among the clients, where each client splits 75% of their local data for training and the remaining 25% for testing.

C.2. Model Heterogeneity

In our experiments, we consider four model heterogeneity settings: HtFE_2, HtFE_3, HtFE_4, and HtFE_9.

1. **HtFE**$_2$ consists of two model architectures: 4-layer CNN [9] and ResNet18 [2].
2. **HtFE**$_3$ consists of three model architectures: ResNet10 [24], ResNet18, and ResNet34 [2].
3. **HtFE**$_4$ consists of four different model architectures: a 4-layer CNN, GoogleNet [14], MobileNet_v2 [12], and ResNet18.
4. **HtFE**$_9$ consists of nine different model architectures: ResNet4, ResNet6, and ResNet8 [24], ResNet10, ResNet18, ResNet34, ResNet50, ResNet101, and ResNet152.

C.3. Introduction to SOTA methods

SOTA Methods in Heterogeneous Federated Learning. In this setting, we compare FedTSP with ten state-of-the-art (SOTA) methods: FedGen [27], FedGH [21], LG-FedAvg [8], FML [13], FedDistill [3], FedKD [18], FedMRL [22], FedProto [15], FedTGP [23], and AlignFed[26].

FedGen trains a generator on the server to generate synthetic features, which are used to align client classifiers. LG-FedAvg aggregates client updates via federated averaging to jointly train a shared classifier across all clients. FedGH trains a global classifier on the server using feature representations uploaded by clients. FML facilitates mutual learning by sharing logits to guide both an auxiliary model and client models. FedDistill aligns clients based on logit matching, while FedKD extends FML by additionally aligning intermediate feature vectors. FedMRL combines features extracted by both an auxiliary model and the local client model during inference. FedProto, FedTGP, and AlignFed leverage global prototypes for client alignment. FedProto directly aggregates client prototypes to obtain global prototypes. FedTGP introduces trainable server-

side prototypes, optimizing their discriminability during training. AlignFed manually designs prototypes and ensures that different class prototypes are uniformly distributed on a hypersphere.

SOTA Methods in General Federated Learning. In this setting, we compare against four general FL methods: FedAvg, FedBABU [10], FedETF [7], and FedFA [25].

FedBABU and FedETF fix the classifier during training, with FedBABU using random initialization and FedETF adopting ETF-based initialization. Since classifier proxies inherently serve as class anchors, these methods can also be viewed as prototype-based approaches. FedFA enforces feature alignment via prototypes, using them to constrain client models and correct classifier predictions during training.

SOTA Methods in Personalized Federated Learning. For PFL, we compare against six methods: FedAvg-FT, FedBABU-FT, FedETF-Per, FedFA-FT, FedPCL [16], and FedPAC [20].

FedAvg-FT, FedBABU-FT, and FedFA-FT fine-tune the classifiers of their respective global models (FedAvg, FedBABU, and FedFA) to obtain personalized models. It is worth noting that fine-tuning the classifier often outperforms full model fine-tuning. FedETF-Per adopts the personalization strategy proposed in its original paper to obtain a personalized model. FedPCL uses prototypes as knowledge carriers to facilitate information exchange. Notably, the original FedPCL paper employs a pre-trained backbone, but for a fair comparison, we train all methods from scratch in this study. FedPAC aligns client features with prototype constraints during training while encouraging classifier sharing among clients with similar data distributions.

C.4. Hyperparameter Settings

General Hyperparameters for HtFL Experiments. For all HtFL experiments, we set the number of communication rounds to 300. At each round, we compute the average accuracy across all clients and report the highest average accuracy as the final result. All experiments are conducted three times, and we report the mean and standard deviation.

Cross-silo scenario: We use 20 clients, all of them participate in each training round. Each client performs 5 local updates per round with a batch size of 100.

Cross-device scenario: Only 20% of clients participate in each training round. Each client performs 1 local update per round with a batch size of 10.

General Hyperparameters for GFL and PFL Experiments. We follow the hyperparameter settings from FedCAC [19]. Communication rounds are set to 500. The total number of clients is 40, with all clients participating in each round. Each client performs 5 local updates per round with a batch size of 100.

Hyperparameter Settings for Baseline Methods.

(1) HtFL methods:

- LG-FedAvg: No additional hyperparameters.
- FedGen: Noise dimension = 32, generator learning rate = 0.1, hidden dimension = feature dimension d, server learning rate = 100.
- FedGH: Server learning rate = 0.01.
- FML: Knowledge distillation hyperparameters: $\alpha = 0.5, \beta = 0.5$.
- FedKD: Auxiliary model learning rate = 0.01, $T_{\text{start}} = 0.95, T_{\text{end}} = 0.95$.
- FedMRL: The representation dimension is set to 128.
- FedDistill: $\gamma = 1$.
- AlignFed: Initial $\lambda = 20$, decayed to $\lambda = 2$.
- FedProto: $\lambda = 1$.
- FedTGP: $\lambda = 1$, margin threshold = 100.

(2) GFL methods:

- FedBABU: No additional hyperparameters.
- FedETF: Feature dimension is set to the number of classes.
- FedFA: $\lambda = 1$.

(3) PFL methods:

- FedAvg-FT, FedBABU-FT, FedFA-FT: After obtaining the global model, each client fine-tunes the classifier for 20 rounds, selecting the highest accuracy as the final result.
- FedETF-Per: Uses the official fine-tuning strategy from its original paper.
- FedPCL: $\tau = 0.07$.
- FedPAC: $\lambda = 1$.

Hyperparameter Settings for FedTSP.

On CIFAR-10, the server training rounds $E_s = 1$, trainable prompt length $m = 1$. On CIFAR-100 and Tiny ImageNet, the server training rounds $E_s = 20$, trainable prompt length $m = 10$. In the cross-silo scenario, we set λ to 7. In the cross-device scenario, we set λ to 1.

C.5. Compute Resources

Experiments are implemented using PyTorch and conducted on 4x NVIDIA V100 GPUs.

D. Performance under Different Model Heterogeneity Scenarios

To further evaluate the adaptability of our method, we examine its performance under varying levels of model heterogeneity. In practice, federated systems often involve clients equipped with different model architectures because of device diversity, computational constraints, or proprietary requirements. Such heterogeneity introduces additional challenges for prototype-based approaches, as discrepancies in feature extractors can distort the shared feature space and hinder the construction of globally consistent prototypes.

We conduct experiments on CIFAR-10, CIFAR-100, and Tiny ImageNet using multiple heterogeneous settings, denoted as HtFE_2, HtFE_3, and HtFE_4, where the subscript indicates the number of distinct feature extractors used across clients. As reported in Table 1, FedTSP consistently outperforms all state-of-the-art baselines across these configurations. The improvements, up to 3.55% on CIFAR-100, demonstrate that FedTSP effectively maintains semantic consistency and robust prototype quality even when client models differ significantly in architecture. This result confirms that the textual semantic guidance provided by FedTSP enables stable cross-model collaboration and alleviates the alignment difficulty inherent in HtFL.

Table 1. Test accuracy (%) of different methods under Dir(0.1) partition on CIFAR-10, CIFAR-100, and Tiny ImageNet across various model heterogeneity settings. The top three results are highlighted as **first**, second, and third, respectively.

	CIFAR-10			CIFAR-100			Tiny ImageNet		
Methods	HtFE_2	HtFE_3	HtFE_4	HtFE_2	HtFE_3	HtFE_4	HtFE_2	HtFE_3	HtFE_4
FedGen	85.18±0.12	84.00±0.06	82.43±0.10	41.56±0.11	40.56±0.33	31.68±0.39	29.28±0.03	29.69±0.08	22.51±0.21
FedGH	83.46±0.24	84.27±0.18	82.23±0.71	41.81±0.14	40.65±0.19	31.04±0.29	29.90±0.05	30.06±0.07	21.91±0.12
LG-FedAvg	85.64±0.05	84.82±0.11	82.52±0.16	43.15±0.06	41.78±0.18	32.64±0.11	30.09±0.05	30.20±0.13	22.64±0.16
FML	87.12±0.05	86.28±0.10	84.62±0.05	42.25±0.09	40.78±0.55	32.95±0.34	29.77±0.15	30.53±0.13	24.06±0.20
FedDistill	87.32±0.03	86.36±0.04	84.39±0.11	43.46±0.09	40.94±0.13	33.05±0.20	31.05±0.07	30.40±0.13	24.69±0.15
FedKD	87.22±0.17	87.27±0.01	84.70±0.15	45.99±0.09	41.88±0.25	35.33±0.19	32.95±0.07	30.70±0.07	25.46±0.13
FedMRL	87.51±0.29	87.28±0.15	85.19±0.29	43.63±0.21	42.16±0.04	36.14±0.16	29.02±0.20	29.30±0.09	25.32±0.17
FedProto	87.36±0.15	86.56±0.15	84.47±0.05	44.99±0.10	40.13±0.15	34.35±0.17	31.70±0.12	30.04±0.07	25.43±0.15
FedTPG	88.06±0.13	86.93±0.05	85.21±0.22	45.81±0.13	40.84±0.10	34.31±0.11	31.74±0.10	30.21±0.13	25.28±0.14
AlignFed	87.04±0.06	85.21±0.12	83.84±0.03	43.43±0.03	42.04±0.02	33.10±0.36	29.80±0.11	29.89±0.56	22.34±0.16
FedTSP-CLIP	88.40±0.06	87.39±0.04	85.57±0.07	48.09±0.37	45.30±0.02	36.95±0.17	31.83±0.03	**33.24±0.03**	25.66±0.11
FedTSP-BERT	**88.52±0.07**	**87.80±0.03**	**85.72±0.03**	**48.17±0.06**	**45.71±0.21**	**37.59±0.23**	**33.17±0.05**	32.06±0.11	**27.04±0.11**

D.1. Performance in Cross-device Scenarios.

To simulate the cross-device setting, we set the client number to 50, 100, and 200, with 20% of clients participating in each round. Considering the limited computational capacity of mobile devices, we set the batch size to 10 and the number of local epochs E_c to 1. As shown in Table 2, our method outperforms all SOTA methods, demonstrating its robustness with increased clients and unstable participation.

E. Effect of Hyperparameters

Robustness to Client Local Epoch E_c. Client local training epochs E_c is a key hyperparameter in FL. To evaluate its impact on different methods, we vary $E_c \in \{1, 5, 10, 20\}$. As shown in the Fig. 1, we make two main observations: (1) Most methods exhibit a certain level of accuracy drop as E_c increases because less frequent communication among clients reduces collaborative effectiveness. (2) Our method remains robust with respect to E_c and significantly outperforms SOTA methods under all settings.

The Impact of λ. λ balances how much each client learns from its local data versus from the server's textual prototypes. We vary λ in the range $[1, 10]$, and the results are illustrated in Fig. 2. We observe the following: (1) As λ increases, both FedTSP-

Table 2. Test accuracy (%) of different methods in cross-device scenarios on CIFAR-100.

Client #	50	100	200
FedGen	41.06±0.09	37.54±0.04	32.27±0.03
FedGH	37.54±0.14	33.82±0.12	24.87±0.14
LG-FedAvg	41.08±0.03	38.48±0.02	33.95±0.16
FML	38.67±0.08	37.31±0.05	33.45±0.11
FedDistill	41.14±0.02	39.31±0.18	34.85±0.16
FedKD	33.99±0.07	32.16±0.08	27.70±0.08
FedProto	40.64±0.10	39.19±0.03	34.89±0.02
FedTGP	41.45±0.12	39.41±0.01	35.24±0.05
AlignFed	41.11±0.01	38.30±0.12	32.97±0.08
FedTSP-CLIP	43.13±0.15	40.94±0.19	35.82±0.12
FedTSP-BERT	44.36±0.16	41.44±0.16	35.83±0.10

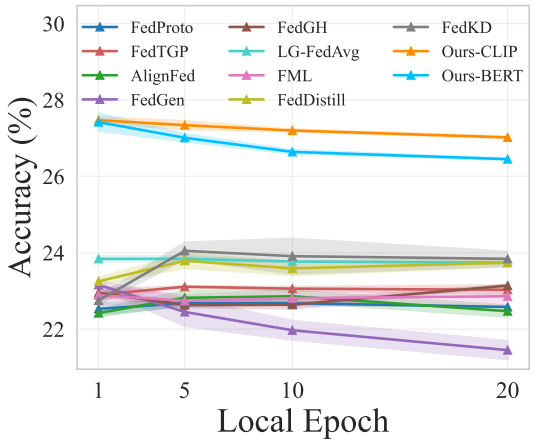

Figure 1. Effect of local epochs E_c on different methods.

BERT and FedTSP-CLIP achieve higher accuracy. This indicates that the textual prototypes on the server effectively provide knowledge beneficial for the client-side image classification tasks. (2) As λ grows larger, the performance gains gradually level off, and the accuracy remains robust for values in the vicinity of $\lambda \in [7, 10]$. This suggests that the method is relatively insensitive to exact λ values within this optimal range.

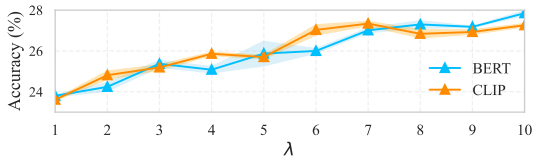

Figure 2. Effect of λ on CIFAR-100.

The Impact of the Trainable Prompt Length m. The trainable prompt length m specifies how many trainable parameters are inserted into the text prompt. It serves as a balance between aligning the server-generated prototypes with the client tasks and retaining the semantic relationships inherent in the text modality. We sample $m \in \{1, 5, 10, 20, 30\}$, and the results are shown in Fig. 3 (left). We can see that near the optimal setting of m, FedTSP exhibits robust performance, indicating its insensitivity to exact m values within this range.

The Impact of the Server Training Rounds E_s. A larger E_s generally leads to a higher degree of alignment between the server-side text prototypes and the client-side image prototypes. We sample $E_s \in \{1, 5, 10, 20, 30, 40\}$. From the results in Fig. 3 (right), we can observe: (1) The performance of FedTSP-CLIP does not vary much with E_s. This is because CLIP already aligns text and image modalities during its pre-training stage. (2) In contrast, FedTSP-BERT's performance consistently improves as E_s increases. Since BERT is not exposed to image data during pre-training, more server training rounds are needed to better align the text prototypes with the image prototypes.

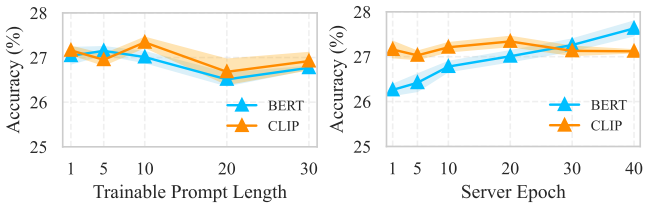

Figure 3. *Left*: The impact of trainable prompt length. *Right*: The impact of server epoch.

Although larger E_s values can further boost the performance of FedTSP-BERT, they also increase the computational load on the server. In our experiments, we therefore set $E_s = 20$ by default.

F. Effect of L2 Regularization

In Section 3.4 of the main paper, we argue that L2-based feature alignment is not suitable for our framework and adopt a contrastive loss for client-side prototype alignment. In this section, we empirically validate this claim. We conduct experiments on CIFAR-10 and CIFAR-100, where FedTSP+L2 denotes a variant of our method that replaces the contrastive alignment loss with an L2 loss. As shown in Table 3, using an L2 loss significantly degrades performance, confirming that simple L2-based alignment is suboptimal for federated prototype learning in our setting.

Table 3. Accuracy (%) of FedTSP and FedTSP+L2 under different Dirichlet partitions on CIFAR-10 and CIFAR-100.

	CIFAR-10			CIFAR-100		
	Dir(0.1)	Dir(0.5)	Dir(1.0)	Dir(0.1)	Dir(0.5)	Dir(1.0)
FedTSP	87.34±0.08	64.76±0.21	58.62±0.07	45.61±0.19	26.92±0.21	20.91±0.18
FedTSP+L2	86.42±0.13	62.60±0.18	55.80±0.22	44.60±0.15	24.90±0.11	18.98±0.24

G. Privacy-Preserving Analysis of Image Prototypes

In the standard FedTSP pipeline, clients upload image prototypes $P_{i,c}^{\mathbf{I}}$ to the server at each communication round (Step 1 in Fig. 2 of the main paper). While prototypes are class-level feature averages rather than raw data, they may still carry information about the underlying training samples. To further mitigate potential privacy leakage from image prototype sharing, we investigate a lightweight quantization-based protection mechanism.

Specifically, before uploading, each client quantizes its local image prototype to b bits per dimension using symmetric uniform quantization. Given a prototype vector $\mathbf{v}$, the quantization is defined as:

$$\alpha = \max(|\mathbf{v}|), \quad s = \frac{\alpha}{2^{b-1}-1}, \quad \hat{v} = \text{clamp}\Big(\Big\lfloor \frac{v}{s} \Big\rceil,\ -(2^{b-1}-1),\ 2^{b-1}-1\Big) \cdot s, \tag{1}$$

where α is the maximum absolute value of the prototype vector, s is the scaling factor, $\lfloor \cdot \rceil$ denotes rounding to the nearest integer, and clamp clips the quantized integer to the b-bit signed range. The server receives the quantized prototypes and dequantizes them (by multiplying the stored integers with s) for subsequent aggregation. Lower bit-widths introduce more quantization noise, making it harder for an adversary to reconstruct the original prototype while also reducing communication overhead.

As shown in Table 4, FedTSP remains robust even under aggressive quantization. With 2-bit quantization, the accuracy degradation is negligible compared to the full-precision baseline across all Dirichlet partitions. These results suggest that prototype quantization can effectively improve privacy protection with minimal impact on model performance, providing a practical and complementary defense mechanism alongside the differential privacy extension described in Section 3.5 of the main paper.

The robustness to quantization can be attributed to two factors. First, image prototypes are obtained by averaging features over all local samples of each class (Eq. (5) in the main paper), which already smooths out individual sample information. Second, on the server side, FedTSP uses prototypes only to update the trainable prompts via contrastive alignment (Eq. (7) in the main paper), which depends on *relative* similarities rather than exact prototype values. This makes the training objective inherently tolerant to small perturbations introduced by quantization.

Table 4. Test accuracy (%) of FedTSP-CLIP under different quantization bit-widths on CIFAR-100 with HtFE9. "base" denotes the full-precision (32-bit) baseline without quantization.

	base	bit=16	bit=8	bit=4	bit=2
Dir(0.1)	45.76	45.78	46.17	45.74	45.50
Dir(0.5)	27.15	26.66	27.17	26.68	26.64
Dir(1.0)	20.85	21.10	20.81	20.58	20.39

H. Robustness to Dynamic Federated Environments

Real-world FL systems often face dynamic conditions, such as evolving data distributions and the arrival of new clients or classes over time. In this section, we evaluate FedTSP's robustness under two representative dynamic scenarios on CIFAR-100 with HtFE9.

H.1. Evolving Data Distributions

In practice, the degree of data heterogeneity across clients may change over time due to shifts in user behavior or data collection patterns. To simulate this scenario, we first train all methods for 100 rounds under a highly heterogeneous setting with Dir(0.1), and then switch the data partition to a less heterogeneous setting with Dir(0.5) for another 100 rounds.

As shown in Fig. 4, during the second phase (rounds 101–200), both FedProto and FedTGP experience noticeable accuracy drops relative to their respective static Dir(0.5) baselines (indicated by dashed lines). This degradation occurs because the prototypes learned during the first phase are biased toward the Dir(0.1) distribution, and adapting to the new distribution requires significant re-learning.

In contrast, FedTSP maintains stable performance and closely matches its static Dir(0.5) baseline throughout the second phase. This stability can be attributed to the fact that FedTSP's text prototypes are derived from a pre-trained language model and are therefore less sensitive to changes in client data distributions. The textual semantic structure provides a robust anchor that remains valid regardless of how local data is partitioned, enabling smoother adaptation when distribution shifts occur.

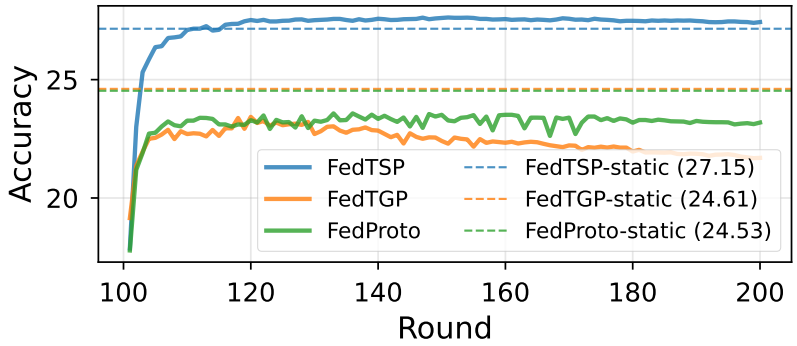

Figure 4. Test accuracy (%) during the second phase (rounds 101–200) after switching from Dir(0.1) to Dir(0.5) on CIFAR-100. Dashed lines indicate the corresponding static Dir(0.5) baselines for each method. FedTSP maintains stable performance close to its baseline, while FedProto and FedTGP exhibit noticeable degradation.

H.2. New Classes and New Clients

Another common scenario in federated systems is the arrival of new clients that introduce previously unseen classes. To evaluate this, we design an open-set experiment on CIFAR-100. We initially train 16 clients on 80 classes for 50 rounds. At round 51, 4 new clients join the federation, each holding data from the remaining 20 classes that were not present during the initial training phase. We then continue training for an additional 100 rounds.

As shown in Fig. 5, we observe distinct behaviors across methods:

- **Impact on base clients (left):** After the new clients join at round 51, both FedProto and FedTGP suffer accuracy drops on the original 16 base clients. This is because the arrival of new classes disrupts the previously learned prototype structure, requiring substantial re-adjustment. In contrast, FedTSP continues to improve on the base clients even after the new clients join, demonstrating its ability to accommodate class expansion without degrading existing knowledge.

- **Convergence on new clients (right):** All methods achieve rapid convergence on the 4 new clients, indicating that the new classes can be effectively incorporated. However, FedTSP achieves higher final accuracy on the new clients, benefiting from the semantically structured text prototypes that provide meaningful initialization for the new classes without requiring extensive prototype re-learning.

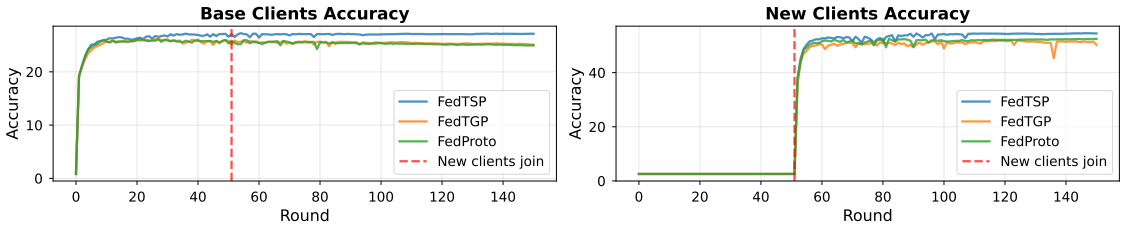

Figure 5. Open-set experiment on CIFAR-100. Left: accuracy on the 16 base clients. Right: accuracy on the 4 newly joined clients. At round 51 (indicated by the vertical dashed line), 4 new clients with 20 previously unseen classes join the federation. FedTSP demonstrates superior stability on base clients and achieves higher accuracy on new clients.

These results demonstrate that FedTSP is well-suited for dynamic federated environments. The text prototypes derived from pre-trained language models serve as stable semantic anchors that can gracefully accommodate both distribution shifts and class/client expansion, without requiring re-initialization or significant re-training.

I. Analysis of Trainable Prompt Dynamics

The trainable prompts introduced in Section 3.3 of the main paper play a central role in bridging the modality gap between the server-side PLM and client-side image models. In Section 4.3 of the main paper, we demonstrated that FedTSP is robust to the choice of LLM used for generating class descriptions (Fig. 5, right). This raises two natural follow-up questions: (1) Do the trainable prompts initialized from different LLMs converge to a similar semantic structure after training? (2) How does the modality alignment evolve over the course of training? In this section, we provide empirical analyses to address both questions.

I.1. Convergence of Semantic Structure Across LLM Initializations

Different LLMs may produce varying textual descriptions for the same set of classes, leading to different initial text prototypes. To investigate whether these differences persist after training, we compare the inter-class similarity matrices of the text prototypes obtained from different LLM initializations, both before and after training.

Specifically, for each LLM initialization, we compute the $C \times C$ cosine similarity matrix among the text prototypes, and then measure the agreement between two similarity matrices using Pearson correlation (measuring linear agreement of similarity values) and Spearman correlation (measuring agreement of similarity rankings).

Table 5. Similarity between inter-class similarity matrices of text prototypes initialized by different LLMs, measured before and after training on CIFAR-100 with HtFE9 and Dir(0.5). Higher values indicate greater agreement in the semantic structure of the learned prototypes.

Comparison	Pearson		Spearman	
	Initial	After training	Initial	After training
Claude vs. GPT-4o	0.7925	0.9878	0.7689	0.8494
Claude vs. Gemini	0.8778	0.9918	0.8507	0.9040

As shown in Table 5, the similarity between the inter-class similarity matrices increases substantially after training across all LLM pairs. For instance, the Pearson correlation between Claude and GPT-4o initializations increases from 0.7925 to 0.9878 after training. This indicates that the trainable prompts effectively converge to a similar semantic topology regardless of the initial LLM choice. The convergence is driven by the contrastive alignment objective (Eq. (7) in the main paper), which aligns text prototypes with client-side image prototypes. Since the underlying visual task is the same, the alignment signal consistently guides the prompts toward a shared semantic structure.

This finding also provides an explanation for the robustness to different LLM choices observed in Section 4.3 of the main paper (Fig. 5, right): even though different LLMs produce different initial descriptions, the trainable prompts adapt during training to recover a consistent semantic structure dictated by the visual task.

I.2. Evolution of Modality Alignment During Training

To understand how the trainable prompts bridge the modality gap during training, we track the text-to-vision retrieval Top-1 accuracy throughout the training process. At each round, we compute the cosine similarity between each text prototype $\overline{P}_c^{\mathbf{T}}$ and all image prototypes $\{\overline{P}_j^{\mathbf{I}}\}_{j=0}^{C-1}$, and check whether the closest image prototype corresponds to the same class. The retrieval accuracy measures the fraction of classes for which the text prototype correctly retrieves its corresponding image prototype.

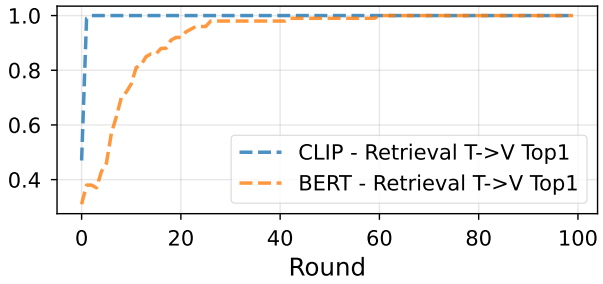

Figure 6. Text-to-vision retrieval Top-1 accuracy during training on CIFAR-100 with Dir(1.0) and HtFE9. As training progresses, the retrieval accuracy for both FedTSP-CLIP and FedTSP-BERT increases and approaches 1.0, indicating that the trainable prompts progressively reduce the modality gap between text and image prototypes.

As shown in Fig. 6, the retrieval accuracy steadily increases during training for both FedTSP-CLIP and FedTSP-BERT, eventually approaching 1.0. This confirms that the trainable prompts are effective in progressively aligning the text prototypes with the image prototypes, thereby bridging the modality gap. Notably, FedTSP-CLIP starts with a higher initial retrieval accuracy than FedTSP-BERT, which is expected since CLIP has already achieved image-text alignment during its pre-training phase. However, FedTSP-BERT catches up quickly, demonstrating the effectiveness of the trainable prompts even for language models that have not been exposed to image data during pre-training.

J. Limitations and Future Work

This paper presents a novel framework for transferring semantic knowledge from the textual modality to client-side visual models in FL. While FedTSP achieves strong performance and effectively integrates textual semantics into prototype-based learning, several aspects warrant further exploration. Currently, class descriptions are generated once at initialization using a fixed prompt, which may not fully capture task-specific nuances or visual discriminative details. Moreover, although multiple fine-grained descriptions are employed to enrich semantic diversity, they may not always align perfectly with the visual semantics required by downstream tasks. In future work, we plan to explore adaptive or feedback-driven prompt generation strategies that dynamically refine textual semantics based on client feedback, thereby further enhancing the flexibility and effectiveness of our framework.

K. Prompts for LLM to Generate Text Descriptions for Each Class and Corresponding Generated Descriptions

In FedTSP, we use the ChatGPT [11] API to generate fine-grained descriptions for each class. Specifically, GPT-4o is used to generate the descriptions in our main experiments, while descriptions generated by Claude and Gemini are further evaluated in the ablation studies. Below are the prompts used in our experiments to generate these descriptions:

K.1. Prompts used in the main experiment (CIFAR-10 example)

```
You are an AI assistant helping to create fine-grained text descriptions for a multi-class image
    classification task in a federated learning setting.

We have a list of classes:
["airplane","automobile","bird","cat","deer","dog","frog","horse","ship","truck"]

For each class in the list, we need you to produce the following outputs:


(1) Class Name and Short Label
```

```
    - A concise label or name for the class.

(2) Fine-grained Text Descriptions (3 versions)
    - Three distinct textual descriptions or prompts that capture different aspects, contexts, or
        attributes of the class.
    - Each description should be a short sentence (1~2 lines) that a vision-language model can use
        to understand key visual/semantic cues of this class.
    - Example aspects: appearance, environment, function, color, shape, typical use, etc.



**Important Requirements and Notes**:
- Keep the format consistent for each class.

Please output the final results in a structured, readable format, grouping the information by each
    class. An example structure:

Class: <class_name>

(1) Short Label:
    - <short label>

(2) Fine-grained Descriptions:
    1) ...
    2) ...
    3) ...

Make sure to repeat the above format for each class.

Now, please generate the requested descriptions and save them in a JSON file.
The JSON file template is:
{
    classname:{
        "Short Label": "Apple",
        "Fine-grained Descriptions":[
        ...
        ],
    },
    ...
}
```

Below are the corresponding descriptions generated by GPT-4o.

```
{
    "airplane": {
        "Short Label": "Plane",
        "Fine-grained Descriptions": [
            "A fixed-wing aircraft with large wings and engine-powered flight for transporting
                passengers or cargo.",
            "Commonly seen at airports, soaring through the sky on jet or propeller propulsion.",
            "Features a streamlined fuselage, prominent wingspan, and tail assembly designed for
                aerodynamic travel."
        ]
    },
    "automobile": {
        "Short Label": "Car",
        "Fine-grained Descriptions": [
            "A four-wheeled motor vehicle commonly used for personal transportation on roads.",
            "Has a compact body with doors, windows, and seats typically arranged in rows.",
            "Powered by an internal combustion or electric engine, designed for efficient on-road
                travel."
        ]
    },
    "bird": {
        "Short Label": "Bird",
        "Fine-grained Descriptions": [
```

```
            "A feathered vertebrate that lays eggs and often takes flight using its wings.",
            "Commonly seen perched on branches or soaring in the sky, with beaks adapted to various
                food sources.",
            "Varies widely in size and color, but typically light-boned for aerial mobility."
        ]
    },
    "cat": {
        "Short Label": "Cat",
        "Fine-grained Descriptions": [
            "A small, domesticated feline known for its agility, retractable claws, and curious
                nature.",
            "Typically has soft fur, upright ears, and a long tail used for balance.",
            "Often kept as a household pet, displaying playful and independent behavior."
        ]
    },
    "deer": {
        "Short Label": "Deer",
        "Fine-grained Descriptions": [
            "A hoofed mammal with slender legs and often antlers on males.",
            "Frequently found in forests or grasslands, grazing on vegetation.",
            "Has a lean body and a short tail, typically moving in cautious herds."
        ]
    },
    "dog": {
        "Short Label": "Dog",
        "Fine-grained Descriptions": [
            "A domesticated canid known for loyalty and companionship.",
            "Often kept as a pet, with varied breeds ranging from small to large in size.",
            "Exhibits social behavior, responsive to training, and known for protective instincts."
        ]
    },
    "frog": {
        "Short Label": "Frog",
        "Fine-grained Descriptions": [
            "A small amphibian with a slimy skin, long hind legs, and the ability to jump great
                distances.",
            "Typically found near ponds, rivers, or wetlands, often emitting croaking sounds.",
            "Has bulging eyes and a wide mouth used for catching insects with its sticky tongue."
        ]
    },
    "horse": {
        "Short Label": "Horse",
        "Fine-grained Descriptions": [
            "A large hoofed mammal with a mane and tail, often ridden or used for work.",
            "Characterized by its strong build, speed, and ability to carry humans or pull loads.",
            "Has a graceful gait and is commonly found in stables, farms, or open pastures."
        ]
    },
    "ship": {
        "Short Label": "Ship",
        "Fine-grained Descriptions": [
            "A large watercraft designed for deep-sea or ocean travel, often carrying passengers or
                cargo.",
            "Built with a sturdy hull and multiple decks, powered by engines or sails.",
            "Operates in open waters, requiring extensive navigation systems and crew operations."
        ]
    },
    "truck": {
        "Short Label": "Truck",
        "Fine-grained Descriptions": [
            "A heavy-duty vehicle designed to transport goods, often featuring a powerful engine and
                 large cargo area.",
            "Commonly seen on highways hauling freight or equipment.",
            "Built with a robust chassis and higher ground clearance for carrying heavy loads."
        ]
    }
}
```

Below are the corresponding descriptions generated by Claude-4-sonnet.


```
{
    "airplane": {
        "Short Label": "Aircraft",
        "Fine-grained Descriptions": [
            "A metallic flying vehicle with wings and engines soaring through the blue sky or clouds
                .",
            "A commercial or military aircraft with streamlined fuselage, typically white or gray in
                 color, designed for air transportation.",
            "A winged machine capable of flight, often seen at airports or in aerial views with
                distinctive tail and wing configuration."
        ]
    },
    "automobile": {
        "Short Label": "Car",
        "Fine-grained Descriptions": [
            "A four-wheeled motor vehicle with doors and windows, commonly found on roads and
                streets for personal transportation.",
            "A passenger car with headlights, bumpers, and various colors, parked or driving on
                asphalt surfaces.",
            "A wheeled vehicle with a enclosed cabin, designed to carry passengers on land with
                steering wheel and automotive features."
        ]
    },
    "bird": {
        "Short Label": "Bird",
        "Fine-grained Descriptions": [
            "A feathered flying animal with wings, beak, and claws, often perched on branches or in
                natural outdoor settings.",
            "A small to medium-sized avian creature with colorful plumage, captured in flight or
                standing positions.",
            "A winged vertebrate animal with distinctive bird features like beaks and talons,
                commonly found in trees, sky, or ground."
        ]
    },
    "cat": {
        "Short Label": "Feline",
        "Fine-grained Descriptions": [
            "A furry domestic animal with pointed ears, whiskers, and a long tail, often in indoor
                household environments.",
            "A small carnivorous mammal with cat-like eyes and graceful posture, displaying typical
                feline behavior and features.",
            "A pet animal with soft fur, four paws, and distinctive cat facial features, commonly in
                 various colors and patterns."
        ]
    },
    "deer": {
        "Short Label": "Deer",
        "Fine-grained Descriptions": [
            "A graceful wild animal with slender legs and often antlers, typically found in forest
                or natural woodland settings.",
            "A brown or tan-colored mammal with a gentle appearance, featuring large ears and alert
                posture in outdoor environments.",
            "A herbivorous animal with characteristic deer silhouette, often shown in profile with
                natural coloring and forest background."
        ]
    },
    "dog": {
        "Short Label": "Canine",
        "Fine-grained Descriptions": [
            "A loyal domestic animal with fur, floppy or pointed ears, and a wagging tail, often in
                home or outdoor settings.",
            "A four-legged pet with canine features like a snout and friendly expression, available
                in various breeds and sizes.",
            "A furry companion animal with dog-typical posture and facial features, commonly
                interacting with humans or in playful poses."
```

```
        ]
    },
    "frog": {
        "Short Label": "Amphibian",
        "Fine-grained Descriptions": [
            "A small green amphibian with bulging eyes and webbed feet, typically found near water
                or on lily pads.",
            "A smooth-skinned creature with a compact body and strong hind legs, capable of both
                swimming and jumping.",
            "A small aquatic animal with a wide mouth and moist skin, often in green coloration with
                 pond or wetland backgrounds."
        ]
    },
    "horse": {
        "Short Label": "Equine",
        "Fine-grained Descriptions": [
            "A large four-legged mammal with a flowing mane and tail, often in fields, stables, or
                being ridden by humans.",
            "A powerful domesticated animal with hooves and elongated face, typically brown, black,
                or white in color.",
            "A majestic riding animal with muscular build and graceful movement, commonly associated
                 with farms, races, or countryside."
        ]
    },
    "ship": {
        "Short Label": "Vessel",
        "Fine-grained Descriptions": [
            "A large watercraft with a hull and superstructure, floating on blue ocean or harbor
                waters for transportation.",
            "A maritime vessel with decks, masts, or smokestacks, designed for sea travel and cargo
                or passenger transport.",
            "A boat-like structure on water with nautical features, often white or gray, navigating
                through seas or docked at ports."
        ]
    },
    "truck": {
        "Short Label": "Heavy Vehicle",
        "Fine-grained Descriptions": [
            "A large motor vehicle with a cargo bed or container, designed for hauling goods and
                materials on roads.",
            "A heavy-duty vehicle with multiple wheels and robust build, typically used for
                commercial transportation and delivery.",
            "A powerful wheeled machine with a cab and loading area, often in industrial or
                construction settings for carrying freight."
        ]
    }
}
```

Below are the corresponding descriptions generated by Gemini-2.5-pro.

```
{
    "airplane": {
        "Short Label": "Airplane",
        "Fine-grained Descriptions": [
            "A winged aircraft designed for air travel, often seen in the sky or at an airport.",
            "A large vehicle with wings and engines that flies through the air.",
            "A fixed-wing aircraft propelled by jet engines or propellers, used for transporting
                passengers or cargo."
        ]
    },
    "automobile": {
        "Short Label": "Automobile",
        "Fine-grained Descriptions": [
            "A four-wheeled road vehicle that is powered by an engine and is able to carry a small
                number of people.",
            "A personal car, typically found on roads, in parking lots, or in a garage, used for
```

```
            daily transport.",
        "A passenger vehicle with a metal body, four wheels, and windows, coming in various
            models like sedans, SUVs, or convertibles."
    ]
},
"bird": {
    "Short Label": "Bird",
    "Fine-grained Descriptions": [
        "A warm-blooded, egg-laying vertebrate distinguished by the possession of feathers,
            wings, and a beak.",
        "A small animal with feathers and wings, often seen perching on tree branches, flying in
             the sky, or foraging for food.",
        "An avian creature of various colors and sizes, known for its birdsong and ability to
            fly."
    ]
},
"cat": {
    "Short Label": "Cat",
    "Fine-grained Descriptions": [
        "A small domesticated carnivorous mammal with soft fur, a short snout, and retractable
            claws.",
        "A common household pet known for its agility, whiskers, and purring, often depicted as
            independent and playful.",
        "A feline creature, often found napping in a sunny spot, grooming itself, or playfully
            chasing toys."
    ]
},
"deer": {
    "Short Label": "Deer",
    "Fine-grained Descriptions": [
        "A hoofed grazing or browsing animal, with branched bony antlers that are shed annually
            and typically grow only on the male.",
        "A wild animal with long legs, a brown or reddish-brown coat, and a white tail, often
            found in forests or fields.",
        "A graceful and slender ruminant mammal, known for its gentle nature and often seen in
            herds."
    ]
},
"dog": {
    "Short Label": "Dog",
    "Fine-grained Descriptions": [
        "A domesticated carnivorous mammal that typically has a long snout, an acute sense of
            smell, and a barking, howling, or whining voice.",
        "A popular pet known as 'man's best friend,' often seen playing fetch, going for a walk,
             or wagging its tail.",
        "A canine with four legs, a tail, and fur, of various breeds and sizes, known for its
            loyalty and companionship."
    ]
},
"frog": {
    "Short Label": "Frog",
    "Fine-grained Descriptions": [
        "A tailless amphibian with a short-bodied, stout build, moist smooth skin, and very long
             hind legs for leaping.",
        "A small, often green or brown creature with bulging eyes, typically found near ponds,
            swamps, or in wet environments.",
        "An amphibian known for its distinct croaking sound and its life cycle that transitions
            from a tadpole in water to a land-dweller."
    ]
},
"horse": {
    "Short Label": "Horse",
    "Fine-grained Descriptions": [
        "A large plant-eating domesticated mammal with solid hoofs and a flowing mane and tail,
            used for riding, racing, and to carry and pull loads.",
        "A majestic animal with a long head and muscular body, often found in stables, pastures,
             or galloping in open fields.",
```

```
            "An equine creature with a strong back and long legs, commonly associated with farming,
                transportation, and equestrian sports."
        ]
    },
    "ship": {
        "Short Label": "Ship",
        "Fine-grained Descriptions": [
            "A large boat for transporting people or goods by sea or other large bodies of water.",
            "A massive vessel with a hull, deck, and superstructure, often seen sailing on the ocean
                 or docked at a port.",
            "A watercraft of considerable size, such as a cargo ship, cruise liner, or naval vessel,
                 navigating through water."
        ]
    },
    "truck": {
        "Short Label": "Truck",
        "Fine-grained Descriptions": [
            "A large, heavy motor vehicle used for transporting goods, materials, or troops.",
            "A commercial vehicle with a cab for the driver and a large cargo area, commonly seen on
                 highways and at loading docks.",
            "A motor vehicle designed to carry heavy loads, characterized by its sturdy frame,
                multiple wheels, and powerful engine."
        ]
    }
}
```

K.2. Prompts used for generating long descriptions in Section 4.3

```
You are an AI assistant helping to create fine-grained text descriptions for a multi-class image
    classification task in a federated learning setting.

We have a list of classes:
["apple","aquarium_fish","baby","bear","beaver","bed","bee","beetle","bicycle","bottle","bowl","boy
    ","bridge","bus","butterfly","camel","can","castle","caterpillar","cattle","chair","chimpanzee
    ","clock","cloud","cockroach","couch","crab","crocodile","cup","dinosaur","dolphin","elephant","
    flatfish","forest","fox","girl","hamster","house","kangaroo","keyboard","lamp","lawn_mower","
    leopard","lion","lizard","lobster","man","maple_tree","motorcycle","mountain","mouse","mushroom
    ","oak_tree","orange","orchid","otter","palm_tree","pear","pickup_truck","pine_tree","plain","
    plate","poppy","porcupine","possum","rabbit","raccoon","ray","road","rocket","rose","sea","seal
    ","shark","shrew","skunk","skyscraper","snail","snake","spider","squirrel","streetcar","
    sunflower","sweet_pepper","table","tank","telephone","television","tiger","tractor","train","
    trout","tulip","turtle","wardrobe","whale","willow_tree","wolf","woman","worm"]

For each class in the list, we need you to produce the following outputs:


(1) Class Name and Short Label
    - A concise label or name for the class.

(2) Fine-grained Text Descriptions (1 version)
    - A detailed paragraph that describes [class].



**Important Requirements and Notes**:
- Keep the format consistent for each class.

Please output the final results in a structured, readable format, grouping the information by each
    class. An example structure:

Class: <class_name>

(1) Short Label:
    - <short label>
```

```
(2) Fine-grained Descriptions:
    1) ...

Make sure to repeat the above format for each class.

Now, please generate the requested descriptions and format them in a JSON style.
The JSON file template is:
{
    classname:{
        "Short Label": "Apple",
        "Fine-grained Descriptions":[
        ...
        ],
    },
    ...
}
```

Example descriptions for the first five classes in CIFAR-100:

```
{
    "apple": {
      "Short Label": "Apple",
      "Fine-grained Descriptions": [
        "An apple is a round fruit, typically red, green, or yellow, with a smooth skin and a crisp,
             juicy interior. It has a slightly sweet flavor and a firm texture, often consumed raw
            or used in cooking and baking."
      ]
    },
    "aquarium_fish": {
      "Short Label": "Aquarium Fish",
      "Fine-grained Descriptions": [
        "Aquarium fish are small, colorful species often kept in home aquariums. They come in a
            variety of shapes and sizes, from bright tetras and guppies to more exotic species like
            angelfish and goldfish. These fish are often kept in water tanks with plants and
            decorative ornaments."
      ]
    },
    "baby": {
      "Short Label": "Baby",
      "Fine-grained Descriptions": [
        "A baby is a young human being, typically from birth to a few years old. Babies are known
            for their small size, delicate features, and rapid development. They communicate through
             crying and cooing and need constant care and attention from their caregivers."
      ]
    },
    "bear": {
      "Short Label": "Bear",
      "Fine-grained Descriptions": [
        "Bears are large mammals with thick fur, sharp claws, and a stocky body. They are typically
            solitary animals, found in forests, mountains, and arctic regions. Bears can be
            omnivores, eating both plants and animals, and are known for their hibernation behavior
            in winter."
      ]
    },
    "beaver": {
      "Short Label": "Beaver",
      "Fine-grained Descriptions": [
        "Beavers are large, nocturnal rodents known for their dam-building skills. They have strong,
             flat tails and sharp teeth for gnawing on trees. Beavers are primarily found near
            rivers or lakes and are famous for creating elaborate dams and lodges out of wood and
            mud."
      ]
    }
 }
```

K.3. Prompts used for generating short descriptions in Section 4.3

```
You are an AI assistant helping to create fine-grained text descriptions for a multi-class image
    classification task in a federated learning setting.

We have a list of classes:
["apple","aquarium_fish","baby","bear","beaver","bed","bee","beetle","bicycle","bottle","bowl","boy
    ","bridge","bus","butterfly","camel","can","castle","caterpillar","cattle","chair","chimpanzee
    ","clock","cloud","cockroach","couch","crab","crocodile","cup","dinosaur","dolphin","elephant","
    flatfish","forest","fox","girl","hamster","house","kangaroo","keyboard","lamp","lawn_mower","
    leopard","lion","lizard","lobster","man","maple_tree","motorcycle","mountain","mouse","mushroom
    ","oak_tree","orange","orchid","otter","palm_tree","pear","pickup_truck","pine_tree","plain","
    plate","poppy","porcupine","possum","rabbit","raccoon","ray","road","rocket","rose","sea","seal
    ","shark","shrew","skunk","skyscraper","snail","snake","spider","squirrel","streetcar","
    sunflower","sweet_pepper","table","tank","telephone","television","tiger","tractor","train","
    trout","tulip","turtle","wardrobe","whale","willow_tree","wolf","woman","worm"]

For each class in the list, we need you to produce the following outputs:


(1) Class Name and Short Label
    - A concise label or name for the class.

(2) Fine-grained Text Descriptions (1 version)
    - A detailed paragraph that describes [class].



**Important Requirements and Notes**:
- Keep the format consistent for each class.

Please output the final results in a structured, readable format, grouping the information by each
    class. An example structure:

Class: <class_name>

(1) Short Label:
    - <short label>

(2) Fine-grained Descriptions:
    1) ...

Make sure to repeat the above format for each class.

Now, please generate the requested descriptions and format them in a JSON style.
The JSON file template is:
{
    classname:{
    "Short Label": "Apple",
        "Fine-grained Descriptions":[
        ...
        ],
    },
    ...
}
```

Example descriptions for the first five classes in CIFAR-100:

```
{
    "apple": {
        "Short Label": "Apple",
        "Fine-grained Descriptions": [
            "A round, red or green fruit with a smooth skin and a crisp, juicy interior."
        ]
    },
    "aquarium_fish": {
```

```
        "Short Label": "Aquarium Fish",
        "Fine-grained Descriptions": [
            "Small, colorful fish typically kept in a glass tank for decorative purposes."
        ]
    },
    "baby": {
        "Short Label": "Baby",
        "Fine-grained Descriptions": [
            "A small, young human, typically under the age of one year."
        ]
    },
    "bear": {
        "Short Label": "Bear",
        "Fine-grained Descriptions": [
            "A large mammal with thick fur, known for its strength and often found in forests."
        ]
    },
    "beaver": {
        "Short Label": "Beaver",
        "Fine-grained Descriptions": [
            "A large rodent with sharp teeth, known for building dams in rivers."
        ]
    }
}
```


References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019. 2

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[3] Eunjeong Jeong, Seungeun Oh, Hyesung Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. *arXiv preprint arXiv:1811.11479*, 2018. 2

[4] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 2

[5] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). *URL http://www. cs. toronto. edu/kriz/cifar. html*, 5, 2010. 2

[6] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 2

[7] Zexi Li, Xinyi Shang, Rui He, Tao Lin, and Chao Wu. No fear of classifier biases: Neural collapse inspired federated learning with synthetic and fixed classifier. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5319–5329, 2023. 3

[8] Paul Pu Liang, Terrance Liu, Liu Ziyin, Nicholas B Allen, Randy P Auerbach, David Brent, Ruslan Salakhutdinov, and Louis-Philippe Morency. Think locally, act globally: Federated learning with local and global representations. *arXiv preprint arXiv:2001.01523*, 2020. 2

[9] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017. 2

[10] Jaehoon Oh, SangMook Kim, and Se-Young Yun. Fedbabu: Toward enhanced representation for federated image classification. In *International Conference on Learning Representations*. 3

[11] OpenAI. Chatgpt: An ai language model, 2023. Accessed: 2024-03-05. 9

[12] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018. 2

[13] Tao Shen, Jie Zhang, Xinkang Jia, Fengda Zhang, Gang Huang, Pan Zhou, Kun Kuang, Fei Wu, and Chao Wu. Federated mutual learning. *arXiv preprint arXiv:2006.16765*, 2020. 2

[14] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 2

[15] Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. Fedproto: Federated prototype learning across heterogeneous clients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8432–8440, 2022. 2

[16] Yue Tan, Guodong Long, Jie Ma, Lu Liu, Tianyi Zhou, and Jing Jiang. Federated learning from pre-trained models: A contrastive learning approach. *Advances in neural information processing systems*, 35:19332–19344, 2022. 3

[17] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 2

[18] Chuhan Wu, Fangzhao Wu, Lingjuan Lyu, Yongfeng Huang, and Xing Xie. Communication-efficient federated learning via knowledge distillation. *Nature communications*, 13(1):2032, 2022. 2

[19] Xinghao Wu, Xuefeng Liu, Jianwei Niu, Guogang Zhu, and Shaojie Tang. Bold but cautious: Unlocking the potential of personalized federated learning through cautiously aggressive collaboration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19375–19384, 2023. 3

[20] Jian Xu, Xinyi Tong, and Shao-Lun Huang. Personalized federated learning with feature alignment and classifier collaboration. In *The Eleventh International Conference on Learning Representations*, 2023. 3

[21] Liping Yi, Gang Wang, Xiaoguang Liu, Zhuan Shi, and Han Yu. Fedgh: Heterogeneous federated learning with generalized global header. In *Proceedings of the 31st ACM International Conference on Multimedia*, 2023. 2

[22] Liping Yi, Han Yu, Chao Ren, Gang Wang, Xiaoxiao Li, et al. Federated model heterogeneous matryoshka representation learning. *Advances in Neural Information Processing Systems*, 37:66431–66454, 2024. 2

[23] Jianqing Zhang, Yang Liu, Yang Hua, and Jian Cao. Fedtgp: Trainable global prototypes with adaptive-margin-enhanced contrastive learning for data and model heterogeneity in federated learning. In *Proceedings of the AAAI conference on artificial intelligence*, pages 16768–16776, 2024. 2

[24] Zilong Zhong, Jonathan Li, Lingfei Ma, Han Jiang, and He Zhao. Deep residual networks for hyperspectral image classification. In *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 1824–1827. IEEE, 2017. 2

[25] Tailin Zhou, Jun Zhang, and Danny H. K. Tsang. Fedfa: Federated learning with feature anchors to align features and classifiers for heterogeneous data. *IEEE Transactions on Mobile Computing*, 23(6):6731–6742, 2024. 3

[26] Guogang Zhu, Xuefeng Liu, Shaojie Tang, and Jianwei Niu. Aligning before aggregating: Enabling communication efficient cross-domain federated learning via consistent feature extraction. *IEEE Transactions on Mobile Computing*, 23(5):5880–5896, 2024. 2

[27] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-Free Knowledge Distillation for Heterogeneous Federated Learning. 2021. 2